Dangerous Action Recognition of Pedestrians with CNN and LSTM

Binghao Wang, Qianyao Yang

March 6, 2017

1 Introduction

For now, we just tried to train the temporal stream with ResNet-50 on KTH dataset and it seemed promising. We have been currently working on MXNet framework. Image processing libraries like OpenCV, PIL(Python Image Library) are also used to extract frames and generate optical flows. Project files and source codes can be found on https://github.com/devaib/dangerous-action-recognition.

2 Preprocessing

Since we can't get optical flow features from videos directly, data preprocessing step is necessary.

2.1 Dataset

KTH a relatively small dataset of 6 classes of actions performed by 25 subjects. The videos come in resolution 160×120 in grayscale and run at 25 frames per second (FPS).

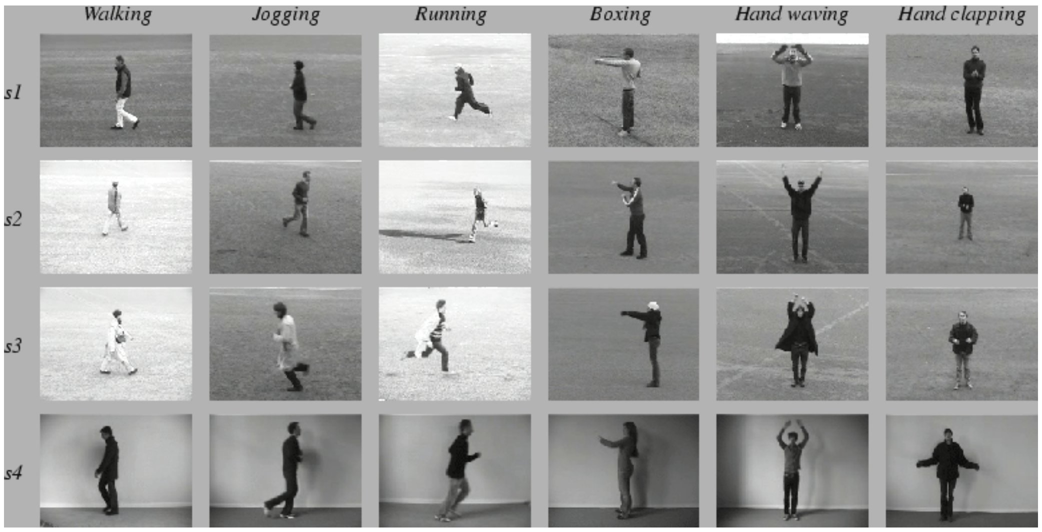

Figure 1: Examples of some frames in the KTH dataset of 6 classes(image from KTH website)

2.2 Data and annotation

Every video clip contains 4 complete actions of a class from a single person. However, since the 4 action are not seamlessly concatenated, we have to remove a couple of frames in between to get clean data. Fortunately, we can easily do that with the sequence.txt file along with the dataset.

Table 1: Example of valid frames in sequence.txt file

Video clip	Frames
person01_boxing_d1	1-95, 96-185, 186-245, 246-360
person02_handclapping_d1	1-102, 103-216, 217-308, 309-398

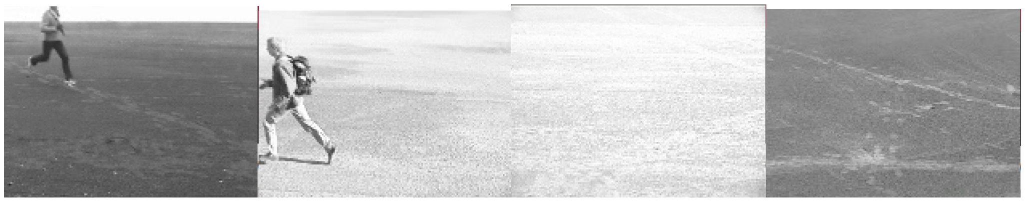
Figure 2: Example of valid(left two) and invalid(right two) frames

We kept the default FPS of 25 and processed every frame in order to gain enough training data. To reduce the expense of training computation at runtime, we resized the videos to 100 x 100 and also take it as the input size of the network.

2.3 Feature extraction

After frame extraction, we generated the vertical and horizontal optical flow of two consecutive frames. For 10 consecutive frames, we stacked these 20(10 x 2) optical flows as a single input data block of size 20 x 100 x 100(channel x width x height). They were cached as Python Pickle for later training process.

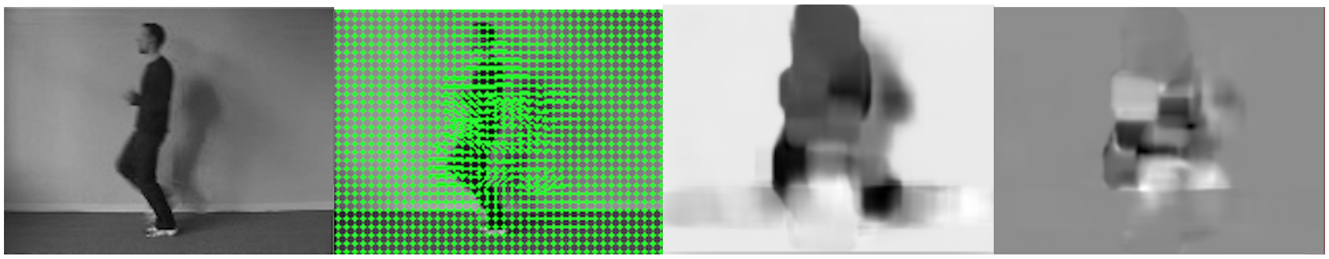
Figure 3: Example of extracted frame, optical flow, and its horizontal and vertical component

3 Model selection

Imagenet was used in the original paper. However, since the video resolution is 160 x 120 in KTH and we didn't want to unsample them, we decided to use the ResNet as it has no restriction of input data size.

4 Implementation

4.1 Training

After several attempts, we picked 0.1 as the learning rate with weight decay of 0.1 and momentum of 0.4.

Because of the large amount of training data, we couldn't load them all in memory at the same time. So we cached the preprocessed data, and loaded each batch from cache while training. This approach cost us significantly time for data I/O.

Following the official guidance, we used person22, 02, 03, 05, 06, 07, 08, 09, 10 as the test set and the rest as trainval(training and validation) set.

After the first epoch, training accuracy is about 90%. It took about 4.5h for 5 epochs, and the training accuracy is over 97%.

4.2 Testing

We haven't finish the evaluation codes but we tried to feed into the network some test data to make sure its validity. Here is what we got from person2 and condition1.

Table 2: Some manual test results on person02 and condition1

ground truth \ prediction	boxing	hand waving	hand clapping	running	walking	jogging
boxing	**.9988**	.0001	.0001	.0000	.0000	.0000
hand waving	.0001	**.9982**	.0012	.0002	.0000	.0002
hand clapping	.0007	.0001	**.9987**	.0000	.0003	.0001
running	.0001	.0006	.0001	.2798	.0006	**.7186**
walking	.9981	.0001	.0001	.0000	**.9981**	.0014
jogging	.0000	.0001	.0000	.0420	.0002	**.9576**

We can see the overall prediction is reasonable except it mislabeled **jogging** instead of **running**. Considering the similarity between jogging and running, it was still a bearable mistake.

5 LRCN

While implementing the temporal stream CNN, we have also been researching on the state-of-the-art techniques of action recognition such as LRCN(Long-term Recurrent Convolutional Network) which benefits from LSTM network. This approach uses CNN for feature extraction and LSTM for sequence learning. It seems to be a more natural way to deal with video sequence since it doesn't require input block only with fixed size.

6 AWS(Amazon Web Services) EC2

We are currently training on a single PC with 32GB RAM and Titan X GPU. Plenty of time were spent on data I/O from disk to memory, and the situation would be even worse when we switch to other larger dataset. We are considering Amazon Web Services EC2.

7 Summary

For the last month, we implemented the temporal stream CNN with ResNet-50 on KTH dataset and achieved a promising result. In the following month, we will finish the evaluation codes and finish the whole two stream structure. We will try to replace optical flow feature with SIFT flow or motion vector to imporve accuracy or make the computation less expensive. Larger dataset like UCF101 and HMDB51 and different network (Inception-7, ResNet-152) will also be tested. After that, we will switch to LRCN to pursue higher accurary.

References

[1] Simonyan, K., & Zisserman, A. (2014). Two-Stream Convolutional Networks for Action Recognition in Videos. arXiv Preprint arXiv:1406.2199, 1–11. http://doi.org/10.1017/CBO9781107415324.004

[2] Zhang, B., Wang, L., Wang, Z., Qiao, Y., & Wang, H. (2016). Real-time Action Recognition with Enhanced Motion Vector CNNs. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2718–2726. http://doi.org/10.1109/CVPR.2016.297

[3] Liu, C., Yuen, J., Member, S., & Torralba, A. (2011). SIFT Flow : Dense Correspondence across Scenes and Its Applications, 33(5).

[4] Donahue, J., Hendricks, L. A., Rohrbach, M., Venugopalan, S., Guadarrama, S., Saenko, K., & Darrell, T. (2015). Long-term Recurrent Convolutional Networks for Visual Recognition and Description, 1–14.